REMARKS

The amendments and remarks presented herein are consistent with those noted in the recent telephone call from Applicant's representative to the Examiner. Accordingly, entry of this amendment and reconsideration of the pending claims is respectfully requested.

The Office Action mailed June 28, 2007, considered and rejected claims 1-28 and 30. Claims 1-2, 9, 13, 16, 19, 23, 25-26, and 30 were rejected under 35 U.S.C. § 103(a) as being unpatentable over *Short* ("Building XML Web Services for the Microsoft®.Net Platform") in view of *Shiran* ("Extending a Class to a Web Service"). Claims 3 and 4 were rejected under 35 U.S.C. § 103(a) as being unpatentable over *Short* in view of *Shiran*, and further in view of *Thomas* (U.S. Publ. No. 2001/0034743). Claims 5 and 7 were rejected under 35 U.S.C. § 103(a) as being unpatentable over *Short* in view of *Shiran*, and further in view of *Pabla* (U.S. Publ. No. 2004/0162871). Claim 6 was rejected under 35 U.S.C. § 103(a) as being unpatentable over *Short* in view of *Shiran* and *Pabla*, and further in view of *Tan* (U.S. Publ. No. 2003/0233360). Claims 10, 11, 20 and 21 were rejected under 35 U.S.C. § 103(a) as being unpatentable over *Short* in view of *Shiran* and further in view of *Tan*. Claim 12 was rejected under 35 U.S.C. § 103(a) as being unpatentable over *Short* in view of *Shiran*, and further in view of *Matsushima* (U.S. Publ. No. 2004/0267808). Claims 17, 18 and 27 were rejected under 35 U.S.C. § 103(a) as being unpatentable over *Short* in view of *Shiran*, and further in view of *Haswell* (U.S. Publ. No. 2005/0193269).[1]

By this paper, claims 1 and 9 have been amended, claims 19-27 and 30 cancelled, and claims 31 and 32 added. Accordingly, following this amendment, claims 1-18, 28, 31 and 32 are pending, of which claims 1, 9, 31 and 32 are the only independent claims at issue.

As previously discussed with the Examiner, and as reflected in the above claims, Applicant's claims are generally directed to embodiments for discovering Web services using a Web service which defines database operation methods in a generic format. For example, independent claims 1 and 9 reflect methods for discovering Web services from the perspectives of a client and Web service, respectively. In such claims, a Web service receives a query from a client and, in response, accesses source code having the definitions compiled into an

[1] Although the prior art status of the cited art is not being challenged at this time, Applicant reserves the right to challenge the prior art status of the cited art at any appropriate time, should it arise. Accordingly, any arguments and amendments made herein should not be construed as acquiescing to any prior art status of the cited art.

intermediate language of a runtime environment, and converts the code of the intermediate language of the runtime environment into a schema-based language for describing Web services in description documents. A description document in that schema-based language is then sent to, and received by, the client. The description document includes, among other things, a generic object class definition and database operation methods defined for the generic object class. The client then compiles the description document into the intermediate language and generates database access requests using these methods by creating an object for a selected type, based on the method for the generic object class.

Newly added claims 31 and 32 recite systems which generally correspond to claims 1 and 9, respectively. Support for the claim amendments and newly added claims can be found throughout Applicant's original application, including at least the disclosure in original paragraph 27.

While *Short* and *Shiran* generally relate to Web services, including their discovery and creation, Applicant respectfully submits that they fail to disclose or suggest Applicant's invention as recited in the above claims. For example, *Short* and *Shiran* fail to disclose or suggest: (i) compiling source code, at a Web service, into an intermediate language of a runtime environment, from a schema-based language, in response to receiving a query from a client; or (ii) in response to receiving a description document, a client converting a description document from a schema-based language into an intermediate language for a runtime environment, as recited in combination with the other claim elements.

For example, *Short* discloses various examples of Web services that may be used to implement different web methods. For instance, *Short* describes a WSDL document which specifies the characteristics of a Web service which implements a calculator, including *Add* and *Subtract* methods, in which the corresponding operation is performed on integers received by the Web service. (Chapter 5, Sections 1 and 2). Additionally, *Short* describes a Web method which accepts a purchase order. For example, the body of a SOAP message includes an instance of a *PurchaseOrder* XML data type and an *AcceptPO* method can be implemented accept the purchase order submitted in the SOAP message, and presumably to store the purchase order in some sort of database. (Chapter 7, Section 6).

Short further recommends that a Web service be debugged at the time that it is compiled, and, accordingly, before a client can run the Web service. (Chapter 11, Section 2). Debugging is

performed at the source-code level, and involves the debugger generating a program database at compile time. (*Id.*). Specifically, the program database, which is generated at compile time, contains a mapping between the Microsift intermediate language (MSIL) and the lines in the source code to which they relate. (*Id.*) The source code may be produced in C#, Visual Basic, Visual Basic .NET, and JScript .NET. (Chapter 6, Section 1), and can then be compiled the first time the Web service is accessed. (*Id.*).

Accordingly, *Short* appears to disclose that when a Web service is accessed the first time, its source code is compiled, and that compiling can generate a mapping from the source code to MSIL. *Short* thus teaches that code is converted by the Web service from source code (e.g., C#) to MSIL. This is directly in contrary to Applicant's claims in which, in response to receiving a query from a client, code already compiled into the intermediate language is accessed and converted into a schema-based language. In other words, *Short* teaches converting into an intermediate language, whereas Applicant claims converting from the intermediate language.

Furthermore, *Short* teaches that a Web service is contained in an .asmx file, and that when the Web service is accessed, it is compiled by a .NET runtime. (Chapter 6, Section§ 1). Accordingly, the *Web service* is compiled *by the server*, and fails to teach wherein the *description document describing* the service is compiled *by the client*, or wherein the client converts a description document into any format, let alone an intermediate language for a runtime environment, as recited in combination with the other claim elements. Indeed, the only mapping in *Short* is between intermediate instructions and source code for the Web service, rather than on a description document describing the service, as claimed.

Applicant respectfully submits that *Shiran* fails to remedy the deficiencies of *Short*. In particular, *Shiran* consists of a single page directed to extending/deriving a class to a Web service. *Shiran* has no discussion of any type directed to discovering Web services, converting between an intermediate language and a schema-based language, or even the use of flag statements or generating, at a client, database operation methods for a selected object type, based on a method for a generic object type.[2]

[2] The Office Action points to eleven lines in *Shiran* as disclosing the concept between a single paragraph of Applicant's original application and concludes that the reference discloses the "claim as a whole." (Office Action, p. 6). In essence, it appears that the Office is contending that a reference which discloses a single element, and the idea behind hit, also discloses the entire claim. Such an assertion is clearly erroneous in that each element of a claim must be examined. This is particularly so when considering that the *Shiran* referenc is a single page which merely discloses that a class is extended to include methods of a WebService class, but which has no description of, among other things: class definitions for a generic object class, a plurality of object type

In view of the foregoing, Applicant respectfully submits that the other rejections to the claims are now moot and do not, therefore, need to be addressed individually at this time. It will be appreciated, however, that this should not be construed as Applicant acquiescing regarding the cited art or the pending application, including any official notice. Instead, Applicant reserves the right to challenge any of the purported teachings or assertions made in the last action at any appropriate time in the future, should the need arise. Furthermore, to the extent that the Examiner has relied on any Official Notice, explicitly or implicitly, Applicant specifically requests that the Examiner provide references supporting the teachings officially noticed, as well as the required motivation or suggestion to combine the relied upon notice with the other art of record.

In the event that the Examiner finds remaining impediment to a prompt allowance of this application that may be clarified through a telephone interview, the Examiner is requested to contact the undersigned attorney at (801) 533-9800.

Dated this 28th day of August, 2007.

Respectfully submitted,

RICK D. NYDEGGER
Registration No. 28,651
JENS C. JENKINS
Registration No. 44,803
COLBY C. NUTTALL
Registration No. 58,146
Attorneys for Applicant
Customer No. 047973

RDN:JCJ:CCN:gd
GD0000002090V001

classes derived from the generic object class, a database directory of Web services, a Web service for discovering Web services, a flag statement identifying an object type, generating a database access request, determining whether an object type is a type identified by a flag statement, creating an object of a specified type using a class definition in a description document, generating a database operation method based on a database operation method of a generic object class, or serializing any object, let alone a created object of the selected object type and including it in a request message.